Instrumente si Tehnici de Baza in Informatica

Semestrul I 2024-2025

Vlad Olaru

Curs 11 - outline

- lucrul in retea
- concepte
  - retea
  - protocol
  - sockets
- exemple de protocoale

Context

- servicii de comunicare inter-proces (IPC)
  - *pipes/FIFOs*

    $ ls –l | more
  - *sockets:* Unix sau TCP/IP

- diferenta de paradigma: *shared memory* vs *message passing*
  - *pipe* – construieste un canal de comunicatie in memoria kernel (partajata) a aceleiasi masini

  => consecinte: sincronizare proceselor care scriu/citesc din *pipe* asigurata de kernel, la fel ca si politica de acces la date FIFO
  - *socket* TCP/IP– construieste un canal de comunicatie intre doua masini diferite legate intre ele printr-o *retea de comunicatie*

  => consecinte: complexitatea operarii retelei de comunicatie (care poate pp implicarea mai multor masini intermediare in efectuarea schimbului de mesaje) impune folosirea unor *protocoale de comunicatie*

Retea

Intranet sau retea

- alcătuit din noduri (calculatoare) numite *host*-uri
- legături fizice: cabluri de retea, canale wireless
- legături logice: calea prin intermediul legăturilor fizice dintre un nod sursă si unul destinatie
- probleme: parcurgere de graf, calea cea mai scurtă, comis-voiajor, retele de flux (network flows)
- o retea poate fi conectată la una sau mai multe retele

Internet: set de retele publice interconectate

Protocol de comunicatie

Modul de comunicare în retea se desfăsoară urmând unul sau mai multe protocoale

- descrise în documente *Request for Comments* (RFC)
- Internet Engineering Task Force (IETF)
- Internet Society (ISOC)
- comitete formate din ingineri și informaticieni
- https://www.rfc-editor.org/retrieve/
- **Obs**: protocolul este o specificatie, nu o implementare

Protocoale de comunicatie

- modelul OSI (Open Systems Interconnection)
  - 7 nivele: *fizic, data link, retea, transport, sesiune, prezentare, aplicatie*
- model alternativ, protocoalele internet (a.k.a. protocoalele Department of Defense, DoD):
  - combina ultimele trei nivele OSI intr-un singur nivel: *aplicatie*
  - combina nivelul fizic si data link intr-un singur nivel: *link*
- conceptual, functioneaza pe principiul *stivei de protocoale*: protocolul de nivel cel mai inalt (*aplicatie*) apeleaza la serviciile protocolului imediat anterior (*prezentare*), samd
  - la nivelul cel mai de jos, datele trimise de protocoalele *data link* sunt trimise peste reteaua fizica
  - la receptie se parcurge calea inversa
- exemple protocoale:
  - data link (retele LAN): Ethernet (CSMA/CD), Token Ring, ATM, Wireless
  - retea: IP
  - transport de date: TCP, UDP, SCTP
  - sesiune: RPC
  - prezentarea datelor: XDR, ASN.1, criptare, compresie
  - aplicatie: DNS, SMTP, HTTP, FTP, SSH, etc
- *socket* : abstractizeaza comunicatia realizata la nivel aplicatie folosind protocoalele de nivel transport ale internetului

Internet Protocol (IP)

- fiecare host identificat printr-una sau mai multe adrese IP
  - versiunea initială (IPv4) foloseste 32-biti (RFC791)
  - versiunea curentă este pe 128-biti (IPv6)
  - IPv4 în continuare cea mai folosită
  - 32 de biti reprezintă 4 octeti grupati separat prin .
  - interval: 0.0.0.0 – 255.255.255.255
  - anumite subintervale sunt folosite pentru adresare privata (adrese nerutabile)
  - restul adreselor publice sunt accesibile de orice nod din retea
  - **ex**: 192.168.1.1, 10.0.0.1, 216.58.214.238, 8.8.8.8

Transport Control Protocol (TCP)

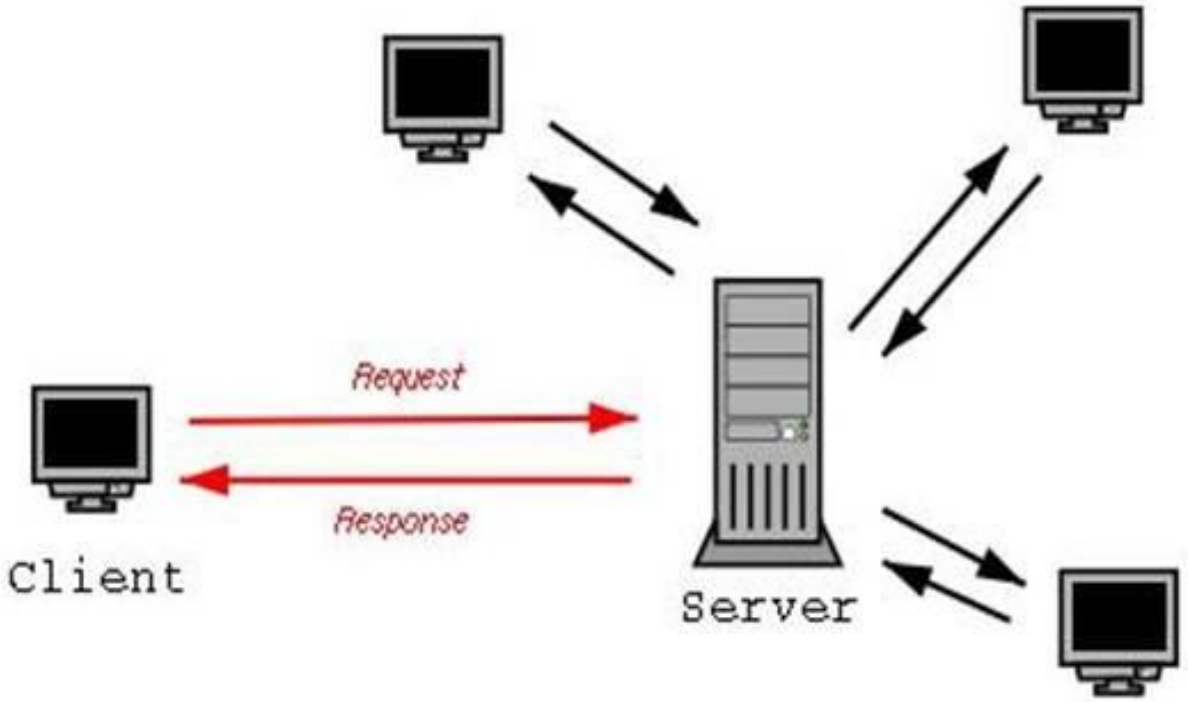
- asambleaza/dezasambleaza mesajele in pachete transmise peste canale de comunicatie eterogene (TCP)
- asigura livrarea pachetelor in ordine la destinatie (TCP)
- rezolva automat probleme de rutare a pachetelor pe cai de comunicatie alternative intre nodurile sursa si destinatie atunci cand anumite masini intermediare nu sunt disponibile (IP)
- asigura retransmisia automata a pachetelor pierdute/defecte (TCP)
- controleaza comunicatia in prezenta congestiei retelei (TCP)
- samd

Domain Name System (DNS)

Numele asociat unei adrese IP

- mai usor de tinut minte pentru utilizatorii umani
- protocol descris în RFC1035
- dacă o adresa IP se schimbă, numele rămâne
- initial toată lumea tinea o agendă locală în `/etc/hosts`
- astăzi procesul e centralizat
- toată lumea foloseste aceiasi agendă
- mai multe *host*-uri, numite *name servers*, care contin o copie a agendei
- serverele principale de la care copiază informatia restul se numesc *root name servers*
- **ex**: `fmi.unibuc.ro` → 193.226.51.6
- comenzi în Unix: `nslookup(1)`, `whois(1)`, `dig(1)`

Arhitectura sistemului distribuit



http://abcnetworking.wikispaces.com

Socket

Comunicarea client-server se face prin *sockets*

- abstractie de nivel sistem de operare
- adresă: adresa IP a unui *host*
- port: intrare a *host*-ului
  - port server: identifica un serviciu (v. */etc/services* pt. well-known ports)
  - port client: identifica procesul client
- adresa si portul formeaza un *endpoint*
- socket: defineste un endpoint
- o conexiune se face intre 2 endpoints folosind protocol comun
- un socket pentru server si unul pentru client
- **ex**: <TCP,192.168.1.6:4444,193.226.51.6:80>

TCP sockets

- canal de comunicatie (o conexiune) intre doua endpoint-uri (adr IP, port)
- garanteaza livrarea corecta si in ordine a mesajelor intre sursa si destinatie
- mesajele pierdute se retransmit automat
- canalul de comunicatie are o semantica asociata de tip stream de octeti (analog *pipe*)
- ca atare, din socket-ul TCP se citesc octeti, nu mesaje intregi !

UDP sockets

- canal de comunicatie *fara garantii* intre doua endpoint-uri
- mesajele (numite in acest caz *datagrame*, ca la protocolul IP) se pot pierde, nu se retransmit automat, nu se livreaza in ordine
- mesajele se citesc in intregime, i.e. o datagrama la un moment dat (spre deosebire de TCP)

Exemple simple protocoale de comunicatie de nivel aplicatie

- *daytime*, *echo*, *time*
- *client*: emulat cu comenzile de tip *telnet*

  $ telnet host port

- *server:* demoni standard accesibili prin superserverul Internet (*inetd/xinetd)*
  - server cu structura particulara care multiplexeaza mai multe servicii diferite
  - serviciile multiplexate configurate in fisiere de configurare aflate in */etc/xinetd.d*
- in general, serverele au fisiere de configurare, clientii au doar parametri de apel in linia de comanda
  - diferenta e dictata de complexitatea diferita a celor doua tipuri de programe
- serviciile controlate cu comanda *service/systemctl*

  $ service <serviciu> <start|stop|reload>

World Wide Web (www)

- creat la CERN și publicat în 1991
- resurse identificate prin Uniform Resource Locators (URL)
- pagini web prezentate prin hypertext
- HyperText Transfer Protocol (HTTP) RFC2616
- servesc pagini web pe portul 80
- acces criptat prin HTTPS pe port 443 (curând implicit)
- **ex**:
  - http://fmi.unibuc.ro:80,
  - http://fmi.unibuc.ro
  - http://fmi.unibuc.ro:443
  - https://fmi.unibuc.ro:443
  - https://fmi.unibuc.ro
- browser Unix: lynx(1), w3m(1)
- crawler Unix: curl(1), wget(1)

Exemplu comunicatie HTTP

- telnet <host> 80
- apoi comanda

  GET / HTTP/1.1 (sau GET /index.html HTTP/1.1)

  Host: <host>

  <cr/lf>

  <cr/lf>

- telnet fmi.unibuc.ro 80

Email – IMAP, POP

- acces la posta
  - Post Office Protocol (POP) RFC1939
  - POP face o copie locală a mesajelor si le sterge de pe serverul de mail
  - Internet Message Access Protocol (IMAP) RFC3501
  - IMAP operează asupra mesajelor direct pe server
  - IMAP poate tine o copie (partială) local
  - porturi: POP 110, POP3S 995, IMAP 143, IMAPS 993
  - mesageria tinută în format mbox sau maildir
  - mbox: tine toate mesajele într-un singur fisier
  - maildir: tine fiecare mesaj într-un fisier separat

Email – SMTP

Expediere mesaje

- Simple Message Transfer Protocol (SMTP) RFC821
- SMTP trimite un mesaj către un destinatar
- protocol vechi, nu cere expeditor → spam, spoofing
- protocol vechi, nu cere autentificare → spam, spoofing
- porturi: SMTP 25, SMTPS
- solutii anti-spam
  - reverse DNS: legătură IP → nume
  - autentificare: utilizator și parolă, criptare
  - Sender Policy Framework (SPF), DKIM (DomainKeys Identified Mail), DMARC (Domain-based Message Authentication, Reporting and Conformace): anti-email spoofing
  - baze de date cu IP-uri de spameri

Exemplu comunicatie SMTP

```
1       $ telnet mail7.imar.ro 25
2       Trying 193.226.4.17...
3       Connected to mail7.imar.ro.
4       Escape character is '^]'.
5       220 mail7.imar.ro ESMTP Postfix
6       HELO <nume host>
7       250 mail7.imar.ro
8       MAIL FROM: <sender's email address>
9       250 2.1.0 Ok
10      RCPT TO: <receiver's email address>
11      250 2.1.5 Ok
12      DATA
13      354 End data with <CR><LF>.<CR><LF>
14      Subject: test
15
16      Acesta este corpul mailului. Textul trebuie incheiat cu un "." singur pe linie, ca mai jos.
17      Subiectul trebuie sa fie pe prima linie dupa raspunsul serverului la comanda DATA si contine
18      cuvantul cheie "Subject:"
19
20      .
21      250 2.0.0 Ok: queued as ABC6B6A022D
22      QUIT
23      221 2.0.0 Bye
24      Connection closed by foreign host.
```

File Transfer Protocol (ftp)

Transfer de fisiere

- protocol vechi din 1971, activ și astăzi RFC959
- autentificare și criptare FTPS
- port separat de comandă (21) și date (20)
- **exemplu**:
  `ftp://[user[:password]@]host[:port]/url-path`
- comandă Unix: `ftp(1)`
- **ex**:

```
$ ftp alex@fmi.unibuc.ro
$ ftp -a fmi.unibuc.ro
$ ftp fmi.unibuc.ro -P 2121
```

Peer-to-peer

Tranfer de fisiere distribuit fără server

- servicii bazate pe Distribuited Hash Tables (DHT)
- toti clientii transmit si primesc date
- mod eficient de transfer pentru date mari
- elimină presiunea de pe un singur server

Secure Shell (ssh)

Protocol sigur de conectare la alt calculator

- specificat în RFC4253, port 22
- implementarea ubicuă OpenSSH creată de grupul OpenBSD
- http://www.openssh.com
- comandă Unix: ssh(1)

```
$ ssh [user[:password]@]host[:port]
```

- autentificare parolă, cheie asimetrică etc.
- o dată autentificat se deschide un terminal pe acel *host* în care puteti începe să dati comenzi
- se poate folosi interfata grafică cu argumentul -X
- folosit peste tot în programare si administrare

Secure Copy (scp)

Copiere prin protocolul SSH

- Secure Copy Protocol (SCP) nu există RFC
- functionează similar cu comanda cp(1)
- are în plus prefixat *host*-ul
- comenzi Unix: scp(1)

```
$ scp hello.c fmi.unibuc.ro:
$ scp hello.c alex@fmi.unibuc.ro:code/
$ scp -r project/ alex@fmi.unibuc.ro:
```

SSH FTP (sftp)

Functionalitate FTP prin protocolul SSH

- SSH File Transfer Protocol (SFTP) nu există RFC
- funcționează similar cu comanda ftp(1)
- profită de autentificare si criptografia de desubt (SSH)
- de obicei tot pe port 22 este servit
- comenzile protocolului SSH diferentiază între tipuri de sesiuni
- comenzi Unix: sftp(1)

```
$ sftp alex@fmi.unibuc.ro
```

Virtual Network Computing (vnc)

- bazat pe protocolul Remote Framebuffer (RFB) RFC6143
  - protocol simplu, bazat pe transmiterea de primitive grafice de la server la client + evenimente de la client la server
- oferă acces la interfata grafică de la distantă
  - functionează cu Windows, macOS si X (sistemul grafic Unix)
- VNC este cea mai cunoscută aplicatie ce implementează si foloseste RFB
- clientul se conectează la server pe portul 5900
- comenzi Unix: vncviewer(1), vncserver(1)

```
$ vncserver :0 – pornește server pe primul display
$ vncviewer fmi.unibuc.ro
$ vncviewer fmi.unibuc.ro:2 – conectează clientul la display-ul 3
$ vncserver -kill :0 – oprește serverul
```

Remote Desktop (rdp)

- Remote Desktop Protocol, proprietar Windows
- functionează similar cu VNC
- oferă în plus acces la sistemul audio si imprimantă
- clientul se conectează la server pe port 3389
- pe Windows clasic e permis un singur utilizator conectat
- pe Windows server se pot conecta mai multi deodată
- comenzi Unix: `rdesktop(1)`, `xfreerdp(1)`

```
$ rdesktop 192.168.1.6
$ rdesktop fmi.unibuc.ro
$ rdesktop -r sound:remote -g 90% 141.85.225.153
```